A Hybrid Swarm Intelligence Algorithm for Multiuser Scheduling in HSDPA

Mehmet E. Aydin[1] Raymond Kwan[1] Cyril Leung[2] Carsten Maple[1] and Jie Zhang[1]

[1] CWIND,University of Bedfordshire, Luton, UK, LU1 3JU
[2] The University of British Columbia, Vancouver, B.C., Canada V6T 1Z4

Abstract. Multiuser scheduling is an important aspect in the performance optimization of a wireless network since it allows multiple users to access a shared channel efficiently by exploiting multiuser diversity. To perform efficient scheduling, channel state information (CSI) for users is required, and is obtained via their respective feedback channels. In this paper, a more realistic imperfect CSI feedback, in the form of a finite set of Channel Quality Indicator (CQI) values, is assumed as specified in the HSDPA standard. A mathematical model of the problem is developed for use in the optimization process. A hybrid heuristic approach based on particle swarm optimization and simulated annealing is used to solve the problem. Simulation results indicate that the hybrid approach outperforms individual implementations of both simulated annealing and particle swarm optimisation.

1 Introduction

An effective method to improve the spectral efficiency in a wireless communication system is the use of adaptive modulation and coding (AMC) [1, 2]. A higher order modulation provides a better spectral efficiency at the expense of a worse error rate performance; a lower rate channel coding generally provides a better error rate performance at the cost of a poorer spectral efficiency. Thus, with a suitable combination of the modulation order and channel coding rate, it is possible to design a set of modulation and coding schemes (MCS), from which a selection is made in an adaptive manner in each transmission-time interval (TTI) so as to maximize system throughput under varying channel conditions. A common requirement is that the probability of erroneous decoding of a Transmission Block should not exceed some threshold value [3].

While the above adaptation is based on the selection of the best MCS, the downlink transmit power is often held constant,[3] as in the HSDPA scheme [3]. In addition, multiple orthogonal channelization codes (multicodes) can be used in transmitting data to a single user, thereby increasing the per-user bit rate and granularity of adaptation [3, 5, 6]. In Wideband Code-Division Multiple Access (WCDMA), such channelization codes are often referred to as Orthogonal

[3] In some cases, the specification stipulates a slight power reduction for a mobile with an exceptionally good channel quality [4].

Variable Spreading Factor (OVSF) codes; the number of OVSF codes per base station (BS) is limited due to their orthogonal property [5]. Thus, in addition to the downlink transmit power, orthogonal code channels are also a scarce resource at the BS.

In exploiting multiuser diversity, a common method employed to increase the network throughput is assigning resources to a user with the largest signal-to-noise ratio (SNR) among all backlogged users (i.e. users with data to send) at the beginning of each scheduling period. However, due to limited mobile capability [4], a user may not be able to utilize all the radio resources available at the BS. Thus, transmission to multiple users during a scheduling period could be more resource efficient. In [7], the problem of downlink multiuser scheduling subject to limited code and power constraints is addressed. It is assumed that the exact path-loss and received interference power at every TTI for each user is fed back to the BS, which would require a large bandwidth overhead.

In this paper, the problem of optimal multiuser scheduling in HSDPA is addressed. The MCSs, numbers of multicodes and power levels for all users are jointly modelled for optimization at each scheduling period, given that only limited CSI information, as specified in the HSDPA standard [4], is fed back to the BS. An integer programming model is proposed in order to provide a globally optimal solution to the multiuser scheduling problem. Due to the complexity of the method, a number of heuristic optimisation algorithms, namely simulated annealing, particle swarm optimization, and a hybrid of these algorithms are subsequently proposed. Following the implementation of simulated annealing [8] and particle swarm optimisation [9], the hybrid method is designed to take benefit of the advantages of both. The experimental results suggest that it may provide a near-optimum performance with significantly reduced practical complexity.

The following two sections describe the multiuser problem domain and a mixed integer programming model. The fourth section provides an introduction to simulated annealing and particle swarm optimization approaches, prior to the implementation of a hybrid algorithm for multiuser scheduling problem. This section is followed by experimental results and discussions of performance before the paper concludes with section six.

2 System Domain

Let P_i denote the default downlink transmit power to user i from a BS, h_i denote the path gain between the BS and user i, and I_i be the total received interference and noise power at user i. The downlink received signal-to-interference ratio (SIR) for user i is given by

$$\gamma_i = \frac{h_i P_i}{I_i},\ i = 1, \ldots, N, \tag{1}$$

where

$$\sum_{i=1}^{N} P_i \leq P_T. \tag{2}$$

Upon receiving the SIR value, γ_i, from user i, the BS decides on the most appropriate combination of MCS and number of multicodes for user i.

If the continuous SIR value γ_i is fed back, an impractically large feedback channel bandwidth would be needed. The specification presented in [4]requires the channel quality information fed back by a mobile, also known as the *channel quality indicator* (CQI), may only be given as a finite number of non-negative integer values $\{0, 1, \ldots, K\}$. The CQI is provided by the mobile via the High-Speed Dedicated Physical Control Channel (HS-DPCCH). Each CQI value maps directly to a maximum bit rate[4] that a mobile can support, based on the channel quality and mobile capability [10], while ensuring that the Block Error Rate (BLER) does not exceed 10%.

While the mapping between the CQI and the SIR is not specified in [4], this issue has been addressed in a number of proposals [11]-[13]. Recently, a mapping has been proposed in which the system throughput is maximized while the BLER constraint is relaxed [14].

Let $\tilde{\gamma}_i = 10\log_{10}(\gamma_i)$ denote the received SIR value at user i in dB and q_i represent the CQI value that user i sends back to the BS via HS-DPCCH. According to [11, 12, 14], the mapping between q_i and $\tilde{\gamma}_i$ can generally be expressed as a piece-wise linear function

$$q_i = \begin{cases} 0 & \tilde{\gamma}_i \leq t_{i,0} \\ \lfloor c_{i,1}\tilde{\gamma}_i + c_{i,2} \rfloor & t_{i,0} < \tilde{\gamma}_i \leq t_{i,1} \\ q_{i,max} & \tilde{\gamma}_i > t_{i,1} \end{cases} \tag{3}$$

where $\{c_{i,1}, c_{i,2}, t_{i,0}, t_{i,1}\}$ are model and mobile capability dependent constants, and $\lfloor . \rfloor$ is the floor function. From (3), it is clear that $\tilde{\gamma}_i$ cannot be recovered exactly from the value of q_i alone due to the quantization. It is important to note that the region $t_{i,0} < \tilde{\gamma}_i \leq t_{i,1}$ is the operating region for the purpose of link adaptation and it should be large enough to accommodate SIR variations encountered in most practical scenarios [5]. In a well designed system, the probability that $\tilde{\gamma}_i$ lies outside this range should be small. As part of our proposed procedure, $\tilde{\gamma}_i$ can be approximated as

$$\tilde{\gamma}_i^{\dagger} = \tilde{\gamma}_i^{(l)} + \left(\tilde{\gamma}_i^{(u)} - \tilde{\gamma}_i^{(l)}\right)\xi, \tag{4}$$

where

$$\tilde{\gamma}_i^{(l)} = \frac{q_i - c_{i,2}}{c_{i,1}}, \tag{5}$$

$$\tilde{\gamma}_i^{(u)} = \frac{q_i + 1 - c_{i,2}}{c_{i,1}}, \tag{6}$$

and ξ follows a uniform distribution, i.e. $\xi \sim U(0, 1)$. Note that this approximation assumes that $\tilde{\gamma}_i$ is uniformly distributed between $\tilde{\gamma}_i^{(l)}$ and $\tilde{\gamma}_i^{(u)}$ for a given value of q_i.

[4] In this paper, the bit rate refers to the transport block size, i.e. the maximum number of bits that a transport block can carry, divided by the duration of a TTI, i.e. 2 ms [5].

When $q_i = 0$ and $q_i = q_{i,max}$, $\tilde{\gamma}_i$ can be simply approximated as $t_{i,0}$ and $t_{i,1}$ respectively, or more generally as $t_{i,0} - \xi_{i,0}$ and $t_{i,1} + \xi_{i,1}$ respectively, with $\xi_{i,0}$ and $\xi_{i,1}$ following certain pre-defined distributions. Finally, the estimated value of γ_i is given by $\hat{\gamma}_i = 10^{\tilde{\gamma}_i^\dagger/10}$. We refer to the mapping from SIR to q_i in (3) as the *forward mapping*, and the approximation of SIR based on the received value of q_i in (4) as the *reverse mapping*.

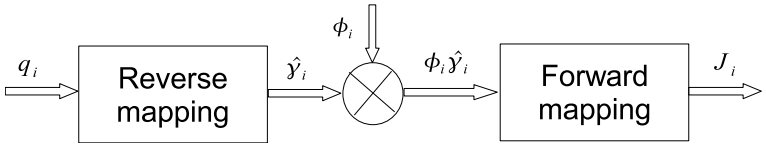

Fig. 1. The conversion process from the received CQI, q_i, from the mobile to the assigned rate index J_i at the base station.

3 Multiuser Scheduling Problem

The CQI feedback value, q_i, from user i corresponds to the rate index that the user requests from the BS, and is associated with a required number of OVSF codes (multicodes) and downlink transmit power. Since the number of multicodes and transmit power are limited, the BS may not be able to simultaneously satisfy the bit rate requests for all users as described by $\{q_i, i = 1, \ldots, N\}$. Thus, given the set $\{q_i, i = 1, \ldots, N\}$, the BS must calculate a set of *modified CQIs*, $\{J_i, i = 1, \ldots, N\}$, for all users by taking into account the power and number of multicodes constraints.

By making use of the *forward* and *reverse* mappings in (3) and (4) respectively, the set of modified CQIs, is given as

$$J_i = \min\left(\max\left(\eta_i(\tilde{\gamma}_i^\dagger, \phi_i), 0\right), q_{i,max}\right), \tag{7}$$

by assigning a power adjustment factor ϕ_i to user i, i.e. $\hat{\gamma}_i \mapsto \phi_i\hat{\gamma}_i$, where

$$\eta_i(\tilde{\gamma}_i^\dagger, \phi_i) = \lfloor c_{i,1}\left(\tilde{\gamma}_i^\dagger + 10\log_{10}\phi_i\right) + c_{i,2}\rfloor, \tag{8}$$

$$0 \le \phi_i \;\le\; 10^{\left(\frac{q_{i,max} - (c_{i,1}\tilde{\gamma}_i^\dagger + c_{i,2})}{10c_{i,1}}\right)}. \tag{9}$$

A summary of the conversion process from the received CQI, q_i, to the final assigned rate index, J_i, is shown in Fig. 1.

The optimal scheduling problem **P1** can be expressed as

$$\mathbf{P1}: \qquad \max_{\mathbf{A},\vec{\phi}} \sum_{i=1}^{N}\sum_{j=0}^{J_i} a_{i,j} r_{i,j} \tag{10}$$

subject to (9), where

$$\sum_{j=0}^{J_i} a_{i,j} = 1, \ \forall i, \tag{11}$$

$$\sum_{i=1}^{N} \sum_{j=0}^{J_i} a_{i,j} n_{i,j} \leq N_{max}, \tag{12}$$

$$a_{i,j} \in \{0, 1\}, \tag{13}$$

$$\sum_{i=1}^{N} \phi_i \leq N, \tag{14}$$

and N_{max} is the maximum number of multicodes available for HSDPA at the BS. The terms $r_{i,j}$ and $n_{i,j}$ correspond to the achievable bit rate and the required number of multicodes associated with CQI value j, where $j \in \{0, 1, \ldots, K\}$, for user i, and are given in [4]. Note that J_i is the maximum allowable CQI value for user i. Depending on code availability, the assigned combination of MCS and the number of multicodes may correspond to a bit rate that is smaller than that permitted by J_i. The constraint in (14) can be obtained by substituting $P_i = \phi_i P_T / N$ into (2). The objective of the above optimization problem is to select the values of the decision variables $\mathbf{A} = \{a_{i,j}\}$ and $\overline{\phi} = \{\phi_i\}$ at each TTI in order to maximize (10), subject to (7)-(9) and (11)-(14).

4 Heuristic Optimization Algorithms

Heuristic optimization methods are based on approximate approaches to solve optimization problems in which a near-optimum solution is guaranteed rather than the global optimum but within much shorter time scheme. Methods can be categorised as either population-based and individual-based approaches; each searches for solutions in different ways; population-based approaches apply their own reproduction operators to generate new solutions, individual-based ones need a neighborhood function. In this paper, we examine the performance of a population-based method, an individual- based heuristic approach and a hybrid implementation. Simulated annealing is presented as an individual-based method and particle swarm optimization as a population-based method.

4.1 Simulated Annealing

Simulated annealing (SA) is one of the most powerful metaheuristics used in optimization for many combinatorial problems. It imposes a probabilistic decision-making process in which a control parameter, called temperature, is employed to evaluate the probability of accepting a non-better neighboring solution of the current solution. The algorithm explores the whole solution space of the problem throughout a simulated cooling process of a given initial (hot) temperature to a

predefined frozen temperature. This process imposes a particular way of searching within the solution space of the problem. Basically, the search with SA is conducted through two nested loops; the outer loop decreases the temperature with a particular cooling schedule, while the inner repeats the search at the same level of temperature. The solutions are altered via some particular neighborhood structures, which are the ways to generate neighbors of the solution undertaken.

Every solution promoted for the next step of the search is a consequence of a probabilistic decision making process. While a newly generated solution that is better than the previous one is accepted, a worse solution can still be accepted with a probability determined by $e^{-\Delta x/\tau_k}$, where Δx is the difference between the performance of current and newly proposed solutions, and τ_k is the temperature level of the k^{th} iteration. The idea behind such an approach is to diversify the search process and to avoid locally optimal solutions. This probabilistic rule is often known as the Metropolis rule in the heuristic optimization literature [15]. Since the probability of promoting a worse solution with the Metropolis rule exponentially decays towards zero, it becomes harder to exploit the perturbation facility, i.e. the way to diverse the solution via moving to a worse neighboring solution, as the temperature decreases.

Evolutionary Simulated Annealing (ESA) is a recently developed SA algorithm enhanced with evolutionary operators [16]. Rather than invoking a single instance of SA with a large number of search iterations, the idea behind ESA is to invoke successive instances of SA, each with a lower number of iterations. While the temperature decreases over the duration of the SA operation, the temperature rises up again every time a new instance of the SA operation is invoked. Such artificially induced fluctuations in temperature allows the solution space to be explored more aggressively, and thereby prevents the solution from being trapped in local optima.

Recently, a comprehensive study on implementing ESA for facility location problems has been reported in [17].

An ESA algorithm is used in solving the aforementioned multiuser scheduling problem. Since the problem is so dynamic and holds high non-linearity, ESA rather than a standard SA is preferred. The mathematical model of the problem undertaken is defined by (7)-(14). The aim is to obtain the most appropriate values for $\overline{\phi}$ and $\mathbf{A}$ such that the objective function (10) is maximized. In this section, the major issue to be tackled is to explore the appropriate values of $\overline{\phi}$ and $\mathbf{A}$ using the ESA searching approach.

The ESA implementation consists of a simulated annealing operator and an initial solution. It is implemented to operate starting with the solution initially provided for G generations[5],where the simulated annealing operator is designed to conduct the search over a certain number of different temperature levels, K, defined as

$$K = \lfloor \log(\tau_{frozen}/\tau_{hot}) / \log\theta \rfloor, \tag{15}$$

[5] The term "generations" is often used to denote the number of iterations in the Evolutionary Computation literature.

where θ is the cooling coefficient, and τ_{hot} and τ_{frozen} are the initial (hot) and frozen temperatures, respectively. At each temperature level, the solution is modified N times by exploiting a *neighborhood structure*, $f(\phi_i, \mathbf{a}_i)$, in order to carry out the exploration within the neighborhood of the solution, where $\mathbf{a_i} = \{a_{i,0}, ..., a_{i,J_i}\},\ i = 1, \ldots, N$.

The states of the problem are constructed based on $\overline{\phi}$ and $\mathbf{A}$, where the former is adopted as the main set of decision variables, while the latter is to be fine-tuned eventually. New solutions are generated by applying $f(\phi_i, \mathbf{a}_i)$, which results in a particular value of ϕ_i, say, ϕ_i'. Correspondingly, $\mathbf{a_i}$ is assigned to the highest available level, and a particular solution i can then be expressed as

$$\mathbf{x_i} = (\phi_1, ...\phi_i', ..., \phi_N, \mathbf{a_1}, ..., \mathbf{a_i}, ...\mathbf{a_N}). \tag{16}$$

In other words, once ϕ_i' is generated, the corresponding J_i is calculated using (7), and a_{i,J_i} is assigned to be 1 while the rest of the elements are set to be 0, i.e. $\mathbf{a_i} = \{\overbrace{0, 0, ..., 0, 1}^{J_i+1}\}$. If constraint (12) is violated, then the non-zero element of $a_{i,j}$ is shifted backward by one position, i.e. $\mathbf{a_i} = \{\overbrace{0, 0, ..., 1, 0}^{J_i+1}\}$. This process is repeated until (12) is satisfied, resulting an updated set of values $\mathbf{a_i'}$. Thus, the new solution can be expressed as

$$\mathbf{x_i'} = (\phi_1, ...\phi_i', ..., \phi_N, \mathbf{a_1}, ..., \mathbf{a_i'}, ...\mathbf{a_N}). \tag{17}$$

The computational complexity of ESA can be considered with regard to the number of users, N, which is the only variable that affects the problem size[6]. The number of generations, G, which is the outermost loop of the algorithm, is fixed at 300, and the temperature levels that controls SA operator is fixed at 200 for any problem size. Thus, the complexity due to these two loops is $\mathcal{O}(1)$. The number of movements at each level of the temperature is proportional to the size of the problem, which results in a complexity of $\mathcal{O}(N)$. Likewise, the neighborhood function checks all details of the solution to ensure that it satisfies all constraints, which corresponds to a complexity of $\mathcal{O}(N)$ in worst case. Thus, the complexity of ESA in the worst case becomes $\mathcal{O}(N^2)$.

4.2 Particle swarm optimization algorithms (PSO)

PSO is one of the population based optimization technique inspired by the social behavior of bird-flocking and fish-schooling. PSO inventors [18, 19] were inspired by such scenarios based on natural processes, explained below, to solve the optimization problems. Suppose the following scenario: a group of birds are randomly searching for food in an area, where there is only one piece of food available and none of them knows where it is, but they can estimate how far it would be. The problem here is "what is the best strategy to find and get that food". Obviously,

[6] In this paper, the number of available rates and the maximum number of codes are assumed fixed.

the simplest strategy is to follow the bird known as the nearest one to the food based on the estimation.

Analogous to this natural process, a population of individual solutions can be adopted as a flock/swarm, where each single solution, called a particle, is considered as a bird/a member of the swarm. The ultimate aim is to explore for the piece of food/optimum solution within a particular area/search space. The achievement of each particle is measured with a fitness value calculated by a fitness function, and a velocity of movement towards the optimum. All particles move across the problem space following the particle nearest to the optimum. PSO starts with an initial population of solutions, which is evolved generation-by-generation towards the ultimate objective(s).

PSO Basics: The pure PSO algorithm builds each particle based on, mainly, two key vectors; position vector, $\mathbf{x_i(t)} = \{x_{i,1}(t), ..., x_{i,n}(t)\}$, and velocity vector $\mathbf{v_i(t)} = \{v_{i,1}(t), ..., v_{i,n}(t)\}$, where $x_{i,k}(t)$, is the position value of the i^{th} particle with respect to the k^{th} dimension ($k = 1, 2, 3, , n$) at iteration t, and $v_{i,k}(t)$ is the velocity value of the i^{th} particle with respect to the k^{th} dimension at iteration t. The initial values, $\mathbf{x_i(0)}$ and $\mathbf{v_i(0)}$, are given by

$$x_{i,k}(0) = x_{min} + rnd_1(x_{max} - x_{min}) \tag{18}$$

$$v_{i,k}(0) = v_{min} + rnd_2(v_{max} - v_{min}) \tag{19}$$

where $x_{min}, x_{max}, v_{min}, v_{max}$ are lower and upper limits of the ranges of position and velocity values, respectively, and rnd_1 and rnd_2 are random numbers uniformly distributed in $[0, 1]$. Since both vectors are continuous, the original PSO algorithm can straightforwardly be used for continuous optimization problems. However, if the problem is combinatorial, a discrete version of PSO needs to be implemented. Once a solution is obtained, the quality of that solution is measured using a cost function denoted by f_i, where $f_i : \mathbf{x_i(t)} \longrightarrow \mathbb{R}$.

For each particle in the swarm, a personal best $\mathbf{y_i(t)} = \{y_{i,1}(t), ..., y_{i,n}(t)\}$, is defined, where $y_{i,k}(t)$ denotes the position of the i^{th} personal best with respect to the k^{th} dimension at iteration t. The personal bests are equal to the corresponding initial position vector at the beginning. Then, in every generation, they are updated based on the solution quality. Regarding the objective function, f_i, the fitness values for the personal best of the i^{th} particle, $\mathbf{y_i(t)}$, is denoted by $f_i^y(t)$ and updated whenever $f_i^y(t+1) \prec f_i^y(t)$, where t stands for iteration and $\prec$ corresponds to the logical operator, which becomes $<$ or $>$ for minimization or maximization problems respectively.

Furthermore, a global best, which is the best particle within the whole swarm is defined and selected among the personal bests, $\mathbf{y(t)}$, and denoted with $\mathbf{g(t)} = \{g_1(t), ..., g_n(t)\}$. The fitness of the global best, $f_g(t)$, can be obtained using

$$f_g(t) = \mathbf{opt}_{i \in N}\{f_i^y(t)\} \tag{20}$$

where **opt** becomes min or max depending on the type of optimization. Afterwards, the velocity of each particle is updated based on its personal best, $\mathbf{y_i(t)}$ and the global best, $\mathbf{g(t)}$ using the following updating rule:

$$v_{i,k}(t+1) = \delta(w_t v_{i,k}(t) + c_1 r_1(y_{i,k}(t) - x_{i,k}(t)) + c_2 r_2(g_k(t) - x_{i,k}(t))) \quad (21)$$

where t is the time index, w_t is the inertia weight used to control the impact of the previous velocities on the current one. It is updated according to $w_{t+1} = \beta\ w_t$, where β is a decrement factor in $[0, 1]$. In (21), δ is a constriction factor which keeps the effects of the randomized weight within the certain range. In addition, r_1 and r_2 are random numbers in $[0, 1]$ and c_1 and c_2 are the learning factors, also known as the social and cognitive parameters. Next, the positions are updated according to

$$x_{i,k}(t+1) = x_{i,k}(t) + v_{i,k}(t). \quad (22)$$

Subsequently, the fitness values corresponding to the updated positions are calculated, and the personal and global bests are determined accordingly. The whole process is repeated until a pre-defined stopping criterion is satisfied.

PSO Implementation: PSO has been applied in various continuous and discrete problems with strong performance [19–21]. The implementation used for this problem is based on a standard PSO as described in subsection 4.2, but without the use of the velocity vector. The reason is due to the difficulty in its initialization and control, and its dispensability in computation. Rather than (22), the rule

$$x_{i,k}(t+1) = x_{i,k}(t) + \Delta x_{i,k}(t) \quad (23)$$

is used, where t is the time index and $\Delta x_{i,k}(t)$ is calculated to be

$$\Delta x_{i,k}(t) = \delta w_{t+1}(c_1 r_1(y_{i,k}(y) - x_{i,k}(t)) + c_2 r_2(g_k(t) - x_{i,k}(t))). \quad (24)$$

The mathematical model of the multiuser scheduling problem undertaken in this implementation is described in Section 3 with the constraints (7)-(14) and the objective function (10). It is a maximization model to optimize two decision variables; $\overline{\phi}$ and $\mathbf{A}$, where $\overline{\phi}$ is a set of positive real numbers identifying the relative power level for each user, and $\mathbf{A}$ is a set of binary variables which identifies the appropriate rate index, j, assigned to user, i, as described in section 3. Regarding these decision variables, the model is a typical mixed-integer programming model. The aim is to obtain the most appropriate values for both sets of variables, such that the throughput of the system is maximized as described in (10). In this case, the problem is to explore the appropriate values of both $\overline{\phi}$ and $\mathbf{A}$ using the technique of PSO.

Similar to the case of ESA implementation, the states of the problem are constructed based on $\overline{\phi}$ and $\mathbf{A}$, where the former is adopted as the main set of decision variables while the latter is to be fine-tuned eventually. New solutions

are generated by applying (23) instead of exploiting a particular neighborhood function. Correspondingly, $\mathbf{a_i}$ is assigned to the highest available level, and a particular solution i can then be expressed as

$$\mathbf{x_i} = (\phi'_1, ...\phi'_i, ..., \phi'_N, \mathbf{a_1}, ..., \mathbf{a_i}, ...\mathbf{a_N}). \tag{25}$$

The way to satisfy the constraints is exactly the same way as in the case of ESA implementation. Thus, the new solution can be expressed as

$$\mathbf{x'_i} = (\phi'_1, ...\phi'_i, ..., \phi'_N, \mathbf{a'_1}, ..., \mathbf{a'_i}, ...\mathbf{a'_N}). \tag{26}$$

It is important to note that the main difference between the two methods for generating new solutions is the number of users tackled each time; where the former considers the change in all users' situation while latter considers a randomly chosen user.

Since PSO is a population based algorithm, in which the population evolves towards a pre-defined objective, (23) is applied to all particles and the whole swarm is updated as a result. In the next step, every particle within the swarm updates the personal best based on the new positions. Subsequently, the best of the whole swarm is determined. As such, the algorithm carries out the search generation-by-generation.

The computational complexity of PSO can be considered with regard to the number of users, N, which is the only variable that affects the problem size.

Since the size of the swarm and the number of generations do not change instance by instance, the complexity corresponds to $\mathcal{O}(1)$. The remaining subject of the complexity is the number of users, N. The algorithm checks with all users to make sure that none of the constraints is violated. Therefore the complexity due to those checks is proportional to the size of the problem, which results in a complexity of $\mathcal{O}(N)$. Thus, in the worst case, the complexity of the algorithm is $\mathcal{O}(N)$.

4.3 Embedding SA into PSO

Since both SA and PSO have their own shortcomings, one possible way to achieve a superior approximate method is to hybridize the methods in a suitable way. Although it is much faster than any exact optimization method, speed of approximation remains the main common shortcoming of SA. Conversely, PSO produces results far faster, however, the corresponding results are less accurate. Therefore, it should be possible to produce more accurate results within a reasonably short time interval, by including the benefits of both methods.

A PSO algorithm was implemented as described in the subsection 4.2 and an SA operator embedded into it, so as to perform a local search on the global best of the current time. The hybrid algorithm operates in a fashion of variable neighborhood search (VNS), in which a local search is refreshed with another, completely different, search procedure. In this case, SA as a local search is diversified with a PSO implementation periodically. Since the SA operator is

re-initialized every iteration, the characteristics of the ESA algorithm are maintained. Thus, this implementation can be recognized as a particular ESA as well as a particular VNS.

The algorithm operates in the following way: A population of solutions is generated, as happens in all population-based algorithms. Then, the PSO algorithm described in subsection 4.2 works to define the personal and the global bests. Once the global best is identified, the SA operator takes this as its initial solution for a local search. The local search with SA takes a small amount of time, and produces a better global best. Once SA finishes, one iteration of the whole algorithm is considered completed. For the next generation (iteration), the PSO procedure works as normal. The algorithm stops once it completes the pre-defined number of generations.

The computational complexity of PSO-ESA is similar to ESA, where the number of users, N, is regarded as the sole subject of complexity. The product of the number of generations, G, and the population (swarm) size, Pop, is the number of steps carried out as the outermost loop of the algorithm. Since neither of them changes instance by instance, the complexity corresponds to $\mathcal{O}(1)$. The remaining subject of the complexity regarding PSO part of the algorithm is the number of users, N. The algorithm checks with all users to make sure that none of the constraints is violated. Therefore the complexity due to those checks is proportional to the size of the problem, which results in a complexity of $\mathcal{O}(N)$. In the worst case, the complexity of PSO is $\mathcal{O}(N)$. On the other hand, the complexity constitutes of SA remains as it happens in the ESA implementation, which was shown in Section 4.1 to be $\mathcal{O}(N^2)$. Thus, the complexity of PSO-ESA in the worst case is $\mathcal{O}(N^2)$.

5 Experimental Results

For illustration purposes, different simulated scenarios for evaluating the performance of the proposed algorithms have been used. The same sets of scenarios have been used in [8] and [9], where SA and PSO approaches are individually implemented for this type of problems. First consider $N = 2$ users with the parameter values in (3) as $t_{i,0} = -4.5$, $t_{i,1} = 25.5$, $c_{i,1} = 1$, $d_{i,1} = 4.5$, and $q_{i,max} = 30$ for $i = 1, 2$; these values are obtained from [11], assuming that the mobiles are of category 10 (i.e. have a wide CQI range) as defined in [4]. Values for $n_{i,j}$ and $r_{i,j}$ are obtained from [4]. Also, let $\{\gamma_i, i = 1, 2\}$ in (1) be the outcomes of Gamma distributed random variables $\{\Gamma_i, i = 1, 2\}$ with probability density functions (PDFs) given by [22]

$$f_{\Gamma_i}(\gamma) = \begin{cases} \left(\frac{\alpha_i}{\overline{\Gamma}_i}\right)^{\alpha_i} \frac{\gamma^{\alpha_i - 1}}{\Gamma(\alpha_i)} \exp\left(\frac{-\alpha_i \gamma}{\overline{\Gamma}_i}\right) & \gamma \geq 0 \\ 0 & \gamma < 0 \end{cases}, \tag{27}$$

where $\Gamma(.)$ is the Gamma function, α_i is the fading figure, and $\overline{\Gamma}_i$ is the mean of Γ_i. The Gamma distribution for the SIR is well-known in the area of wireless communications [22]. The parameter values are listed in Table 1. Let $\mathbf{\Gamma} =$

$\{\Gamma_1, \quad \Gamma_2\}$, and let the aggregate transport block size (TBS), T, per Transmission Time Interval (TTI) be

$$T = \sum_{i=1}^{N} \sum_{j=0}^{J_i} a_{i,j} r_{i,j}. \tag{28}$$

The problem instances generated as 3 different scenarios with a global optimization algorithm have been solved using the branch-and-bound method based on the linearized model as described in [8], which is named here as the Joint Global Optimum (JGO). Then, a simple greedy (SG) algorithm is used to draw a minimum level of solution quality. This SG simply allocates resources to the users in decreasing order of their estimated SIR values, $\hat{\gamma}_i$: the user with the highest $\hat{\gamma}$ is first allocated as much resource as it can possibly use. Subsequent users are allocated in the same way until all resources are exhausted. Investigations for better solution quality within a reasonable time frame were performed using two heuristic approaches; ESA, a simulated annealing algorithm and a PSO implementation. Finally, the same problem instances are solved using a hybrid algorithm of PSO and SA, where PSO works as normal, however a SA operator picks the global best of each generation and improves it for a rather short period. All corresponding descriptions are provided in Section 4. The parameters of PSO are set as follows: both the size of the swarm and the number of generations are given as 100. The other PSO-related parameters are chosen to be $\delta = 1$, $w_0 = 0.98$, $\beta = 0.99998$ and $c_1 = c_2 = 1$. Likewise, the parameters for SA operator are set as follows: stopping temperature, $\tau_{frozen} = 0.01$, highest temperature, $\tau_{hot} = 100$, and the cooling coefficient, $\theta = 0.955$ while the inner iteration is single.

Table 1. Parameters used in generating the benchmark scenarios

Scenario	User i	$\overline{\Gamma}_i$ (in dB)	α_i
I	1	5	6.5
	2	6	3
II	1	12	6.5
	2	10	3
III	1	17	6.5
	2	16	3

The results are presented in Fig. 2 as the cumulative distribution function (CDF) of T for the three different scenarios defined in Table 1. As can be seen, the graphs show that the performances of ESA, PSO, the hybrid algorithm (PSO-ESA) and JGO are similar, and are consistently superior to that of SG. Table 2 summarizes the comparison of the average gains, i.e. $E_{\mathbf{\Gamma}}[T]$, of ESA, PSO, PSO-ESA and JGO over SG for the three scenarios. As is seen, the approaches have

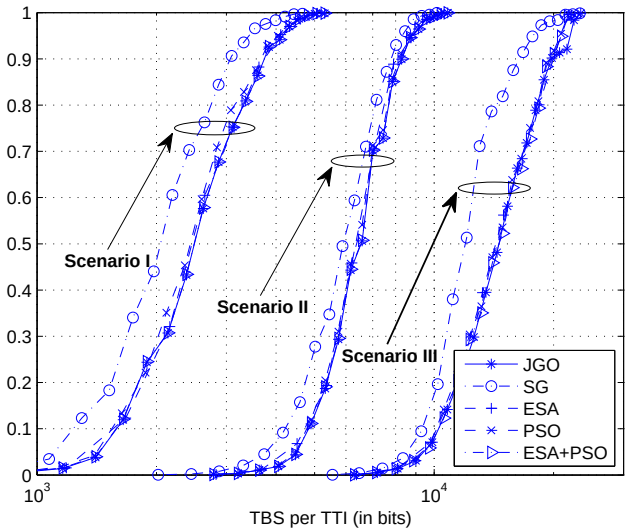

Fig. 2. Performances of different algorithms for 3 scenarios described in Table 1 in CDF of the total number of multicodes at each TTI.

made 17-21% gain for Scenario I and III, while achieving 7-8% for Scenario II. These results show that a good throughput improvement can be achieved with any of these approaches over SG, and that the performances of heuristics are very similar to that of JGO, which provides the global optimum since the graphs almost overlap in Fig. 2.

Table 2. The average gain of the approaches for three scenarios.

Scenario	ESA (%)	PSO (%)	PSO-ESA (%)	JGO (%)
I	20	18	20	21
II	8	6	7	8
III	17	17	18	20

Since the comparative results provided for three scenarios do not help to distinguish the performance of each heuristic algorithm,larger sized problem instances with from 2-8 number of users have been generated . Fig. 3 shows the CDFs of T corresponding to the SG, ESA, PSO and PSO-ESA approaches for 3 different numbers of users, (2, 4 and 8) over 2000 simulation instances. For each user, α=5 and $\overline{\Gamma}$= 8.45 dB. Due to the excessively long computational time, JGO was not considered, but the performances of the heuristics are expected

to be reasonably close to the best performance. Meanwhile, Fig. 4 presents the average TBS per TTI as a function of the number of users for various schemes. As is clearly seen, the performance of heuristics can be easily distinguished on growing size of the problems with the number of users. Fig. 3 indicates that the performances of heuristics except SG are almost the same when the number of users is 2 since the graphs almost overlap. In the 4-user case, it can be seen that the results for PSO-ESA and ESA are similar, and are slightly better than that of the PSO. The achievement with PSO-ESA becomes further evident in the case of 8 users as the graphs of each heuristic approach clearly apart. As is shown in Fig. 4, the results for all heuristics are similar for the 2-user case. However, as the number of users increases, the performance of different heuristics become more apparent; the performances between ESA and PSO-ESA are similar. However, it can be seen that the PSO-ESA outperforms ESA when the number of users is greater than 4. Furthermore, SG performs consistently worse than any other heuristic for any number of users.

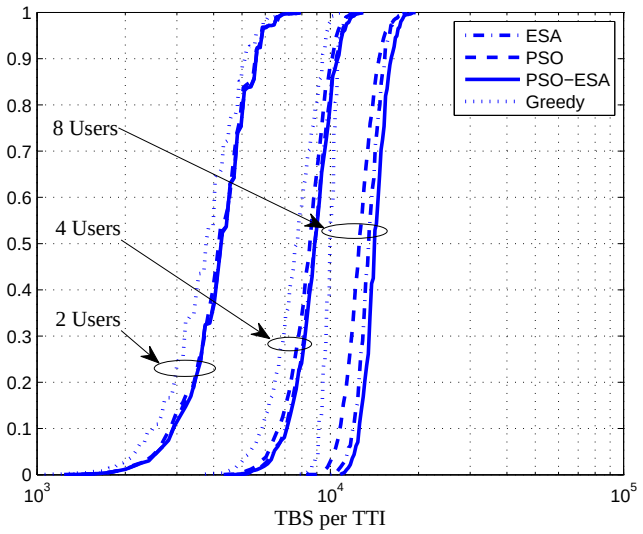

Fig. 3. CDF of the total number of multicodes at each TTI for 2, 4 and 8 user problems.

The complexity of each of heuristic approach is briefly discussed in Section 4, and is shown to be on the order of $\mathcal{O}(N^2)$, $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ for ESA, PSO and the hybrid algorithem, respectively. Note that such complexity representation corresponds to the worse case. Table 3 shows the normalized CPU time, i.e. the CPU time relative to that of the 2-user case, for each of the four algorithms, based on 25 simulation instances. The idea is to reveal the trend behind the

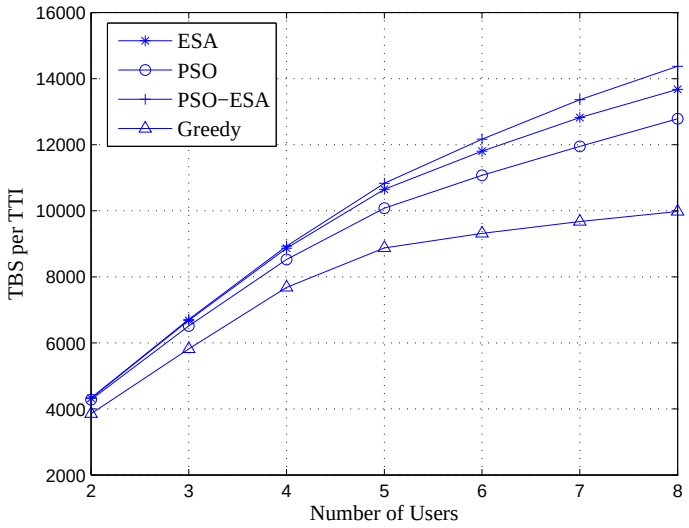

Fig. 4. The average TBS at each TTI for 2 - 8 user problem cases. For each user, α=5 and $\overline{I}$= 8.45 dB.

computational complexity of the algorithms. It can be seen that the growth in complexity for JGO appears to be of an exponential nature, while those for the other three algorithms are linear, and are very similar. The growth of ESA's relative CPU time seems steeper than the other two. Such an observation agrees with the complexity analysis for each approach as presented in Section 4, although the trends for ESA and PSO-ESA remain much milder than the worst cases considered in each corresponding sub section.

Table 3. Performances of all three approaches on growth of problem size.

# of Users	SG	PSO	ESA	PSO-ESA	JGO
2	1.00	1.00	1.00	1.00	1.00
3	1.32	1.33	1.82	1.47	5.10
4	1.76	1.76	2.82	2.06	49.80
5	2.11	2.00	4.11	2.65	961.09

Figure 5 presents a comparative results with respect to CPU time elapsed using each algorithm for 2, 3, 4 and 5 users. As can bee seen, ESA takes much longer than the other two while PSO is much faster. The hybrid method remains

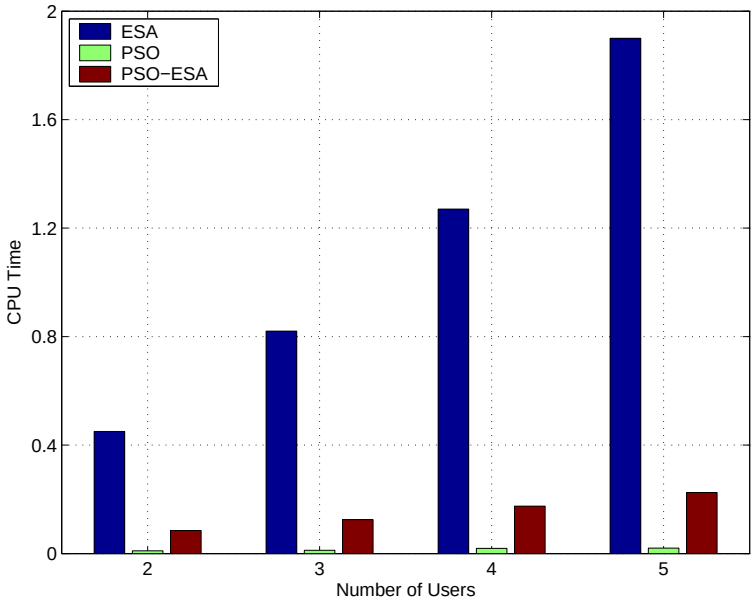

Fig. 5. Averaged CPU time elapsed to solve problems of 2, 3, 4 and 5 user cases

as the intermediate as expected, but much closer to PSO than ESA. This provides a good compromise with the performance with respect to the solution quality.

6 Conclusions

Three methods for allocating resources to multiple users in the context of HSDPA were proposed in conjunction with a problem formulation which adopts the channel feedback scheme specified in the WCDMA standard. Simulation results suggest that the proposed optimization methods can provide a substantial throughput improvement over a greedy approach. ESA performs much better with respect to solution quality, while PSO solves the instances far faster but with poorer quality of solutions. The solutions from the hybrid of PSO and ESA algorithms is found to be much closer to that of the optimal algorithm than both ESA and PSO for all user cases. The advantage of the proposed methods increases with the number of users. Meanwhile, the complexity of PSO-ESA is not worse than ESA, which has complexity of $O(N^2)$. Although the theoretical complexity remains $O(N^2)$, practically it materializes to be rather linear.

References

1. Alouini, M.S., Goldsmith, A.J.: Adaptive Modulation over Nakagami Fading Channels. In: Wireless Personal Communications. Volume 13. Kluwer Academic Pub-

lishers, Netherlands (2000) 119–143
2. Falahati, S.: Adaptive Modulation systems for Predicted Wireless Channels. IEEE Transactions on Communications **52**(2) (Feb. 2004) 307 – 316
3. Holma, H., Toskala, A., eds.: HSDPA/HSUPA For UMTS High Speed Radio Access for Mobile Communications. John Wiley & Sons (2006)
4. : Universal Mobile Telecommunications Systems (UMTS); Physical Layer Procedures (FDD). Technical Specification 3GPP TS25.214, 3rd Generation Partnership Project (2007)
5. Holma, H., Toskala, A., eds.: WCDMA for UMTS Radio Access for Third Generation Mobile Communications. 3 rd edn. John Wiley & Sons (2004)
6. Lee, S.J., Lee, H.W., Sung, D.K.: Capacities of Single-Code and Multicode DS-CDMA Systems Accommodating Multiclass Service. IEEE Trans. on Vehicular Technology **48**(2) (March 1999) 376 –384
7. Kwan, R., Leung, C.: Downlink Scheduling Schemes for CDMA Networks with Adaptive Modulation and Coding and Multicodes. IEEE Transactions on Wireless Communications **6**(10) (October 2007) 3668 – 3677
8. Kwan, R., Aydin, M.E., Luang, C., Zhang, J.: Multiuser scheduling in High Speed Downlink Package Access. IET Communications **3**(8) (2009) 1363–1370
9. Aydin, M.E., Kwan, R., Luang, C., Zhang, J.: Multiuser scheduling in HSDPA using particle swarm optimisation. In: Lecture Notes in Computer Science, Evo-COMNET'09. Volume 5484. (April 2009) 71–80
10. : Universal Mobile Telecommunications Systems (UMTS); UE Radio Access Capabilities. Technical Specification 3GPP TS25.306, 3rd Generation Partnership Project (2007)
11. Motorola, Nokia: Revised CQI Proposal. Technical Report R1-02-0675, 3GPP RAN WG1 (April 2002)
12. Brouwer, F., de Bruin, I., Silva, J.C., Souto, N., Cercas, F., Correia, A.: Usage of Link-Level Performance Indicators for HSDPA Network-Level Simulation in E-UMTS. In: Proc. of International Symposium on Spread Spectrum Techniques and Applications (ISSSTA), Sydney, Australia (September 2004)
13. Ko, K., Lee, D., Lee, M., Lee, H.: Novel SIR to Channel-Quality Indicator (CQI) mapping method for HSDPA System. In: Proc. of IEEE Vehicular Technology Conference (VTC), Montreal, Canada (September 2006)
14. Freudenthaler, K., Springer, A., Wehinger, J.: Novel SINR-to-CQI Mapping Maximizing the Throughput in HSDPA. In: Proc. of IEEE Wireless Communications and Networking Conference (WCNC), Hong Kong, China (March 2007)
15. Chu, K.: Optimal Parallelisation of Simulated Annealing by State Mixing. PhD thesis, State University of New York at Stony Brook, Stony Brook, NY (2001)
16. Aydin, M.E., Fogarty, T.C.: A Distributed Evolutionary Simulated Annealing Algorithm for Combinatorial Optimisation Problems . Journal of Heuristics **10**(3) (May 2004) 269 – 292
17. Yigit, V., Aydin, M.E., Turkbey, O.: Solving large-scale uncapacitated facility location problems with evolutionary simulated annealing . International Journal of Production Research **44**(22) (November 2006) 4773 – 4791
18. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proc. of the 6th Int. Symposium on Micro-Machine and Human Science. (1995) 39 – 43
19. Kennedy, J., Eberhart, R., Shi., Y.: Swarm Intelligence. Morgan Kaufmann, San Mateo, CA, USA (2001)
20. Salman, A., Ahmad, I., Al-Madani, S.: Particle Swarm Optimization for Task Assignment Problem . Microprocessors and Microsystems **26** (2003) 363 – 371

21. van denBergh, F., Engelbecht, A.: Cooperative Learning in Neural Net-works Using Particle Swarm Optimizers. South African Computer Journal **26** (2000) 84 – 90.
22. Simon, M.K., Alouini, M.S.: Digital Communication over Fading Channels. 2 edn. John Wiley & Sons (2005)